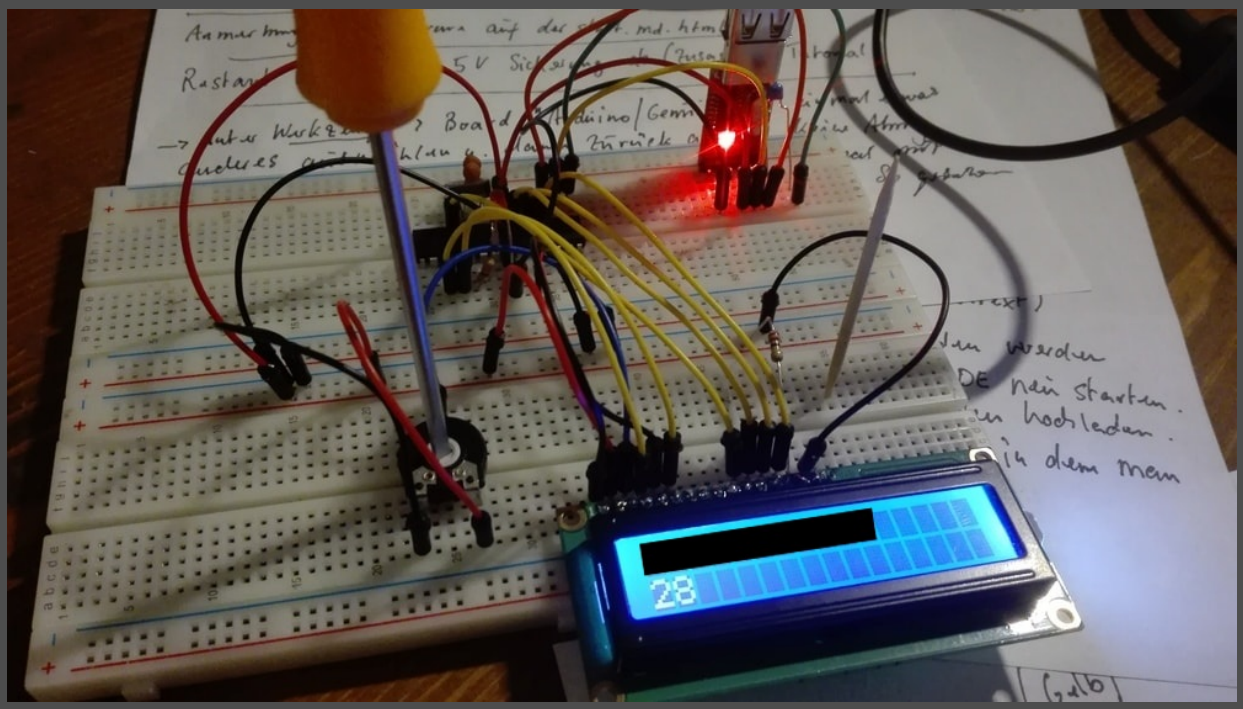


Cyberdeck V3

Preface

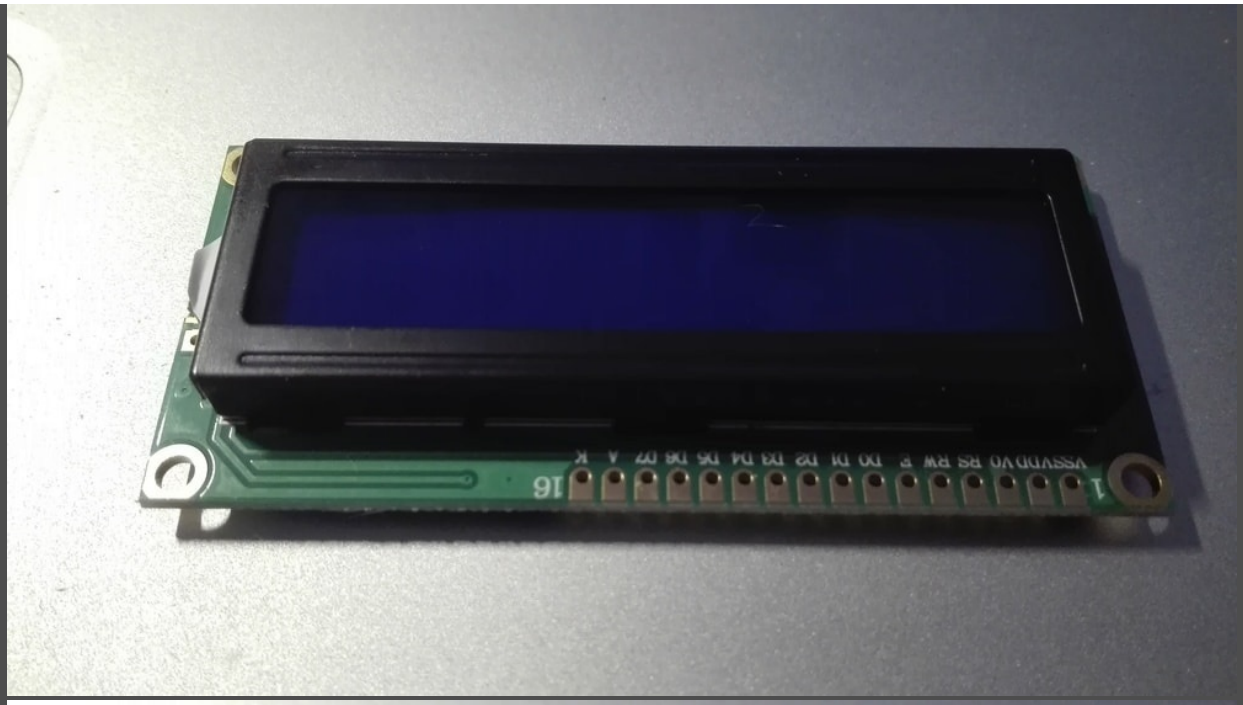
In November 2017 we took the first step towards our own cyberdeck by purchasing an Arduino Microcontroller and testing its functionality in a first attempt. In this documentation we will connect an LCD, i.e. a liquid crystal display, to display simple information. I have oriented myself on the design of learning computers, such as the LC80, POLY880 CP1. These are classics, new models are primarily known through VTECH. My plan is to display information later.

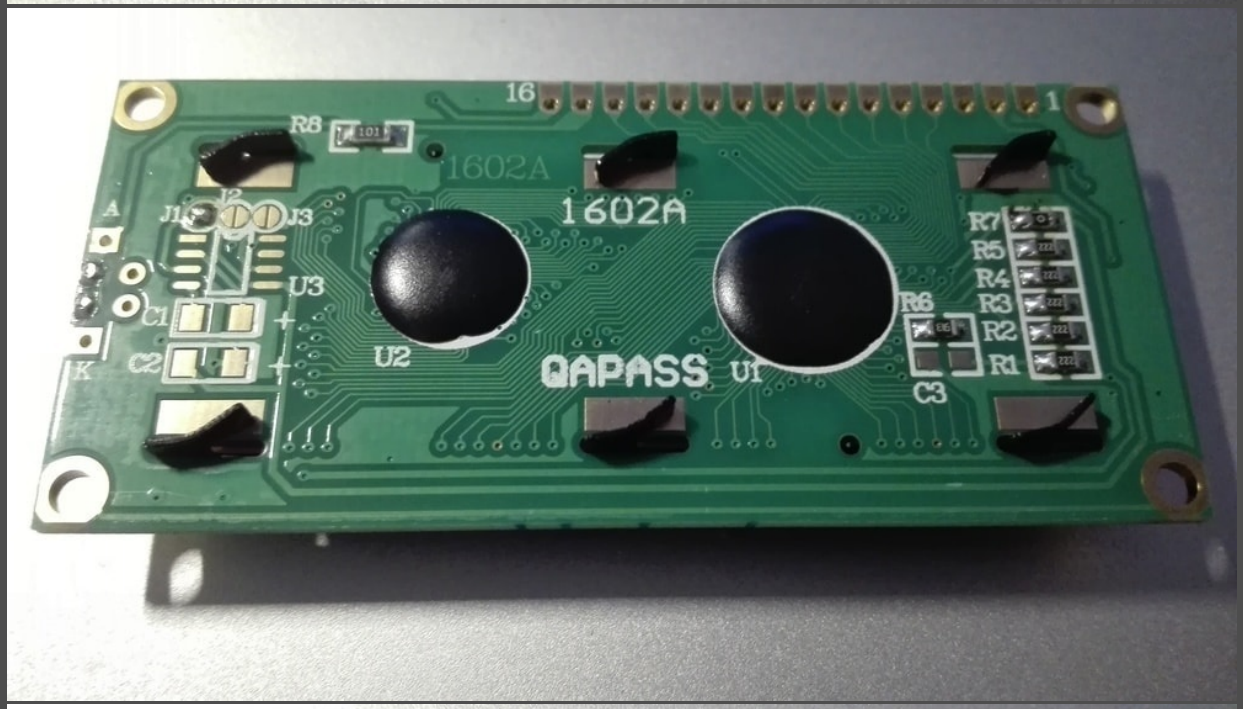
It should then go in the direction of a very simple computer. Of course you ask yourself why you can't develop a Raspberry Pi right away. In the Heinz-Nixdorf-Computer-Museum this was shown to me very well, because the whole technology of computer science available to us is based on the basic ideas of information storage, processing, transmission and presentation. For example, Ada Lovelace, George Boole, Charles Babbage and Gottfried Wilhelm Leibniz can be described as pre:informaticians, since they have made important achievements in this field. The history of computers does not start in the 21st century.

Research

During my research I focused on what liquid crystal displays are available and which are the easiest to connect to an Arduino Uno, respectively to an Atmega328P microcontroller. I first watched some videos that go in this direction, but then orientated myself on a tutorial by Arduino. My goal of spending as little money as possible on an ad was easily achieved with the HD44780. It was also so easy to set up that it could achieve results quickly.

Materials

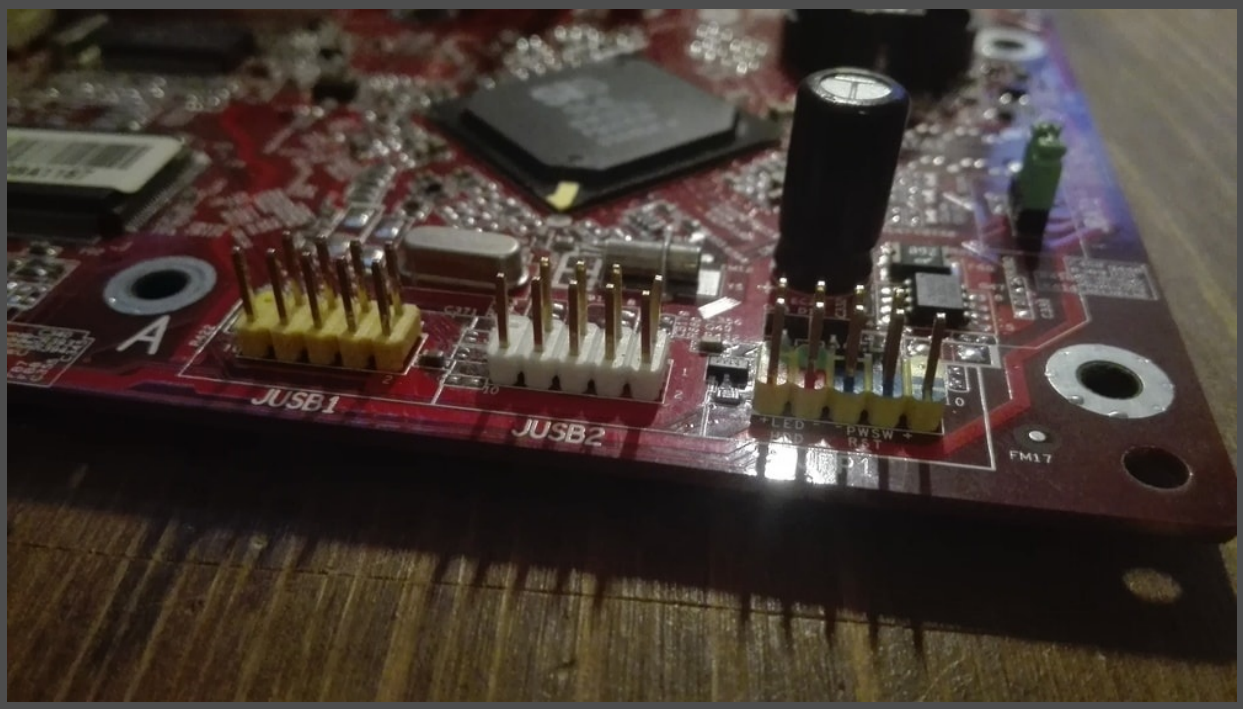




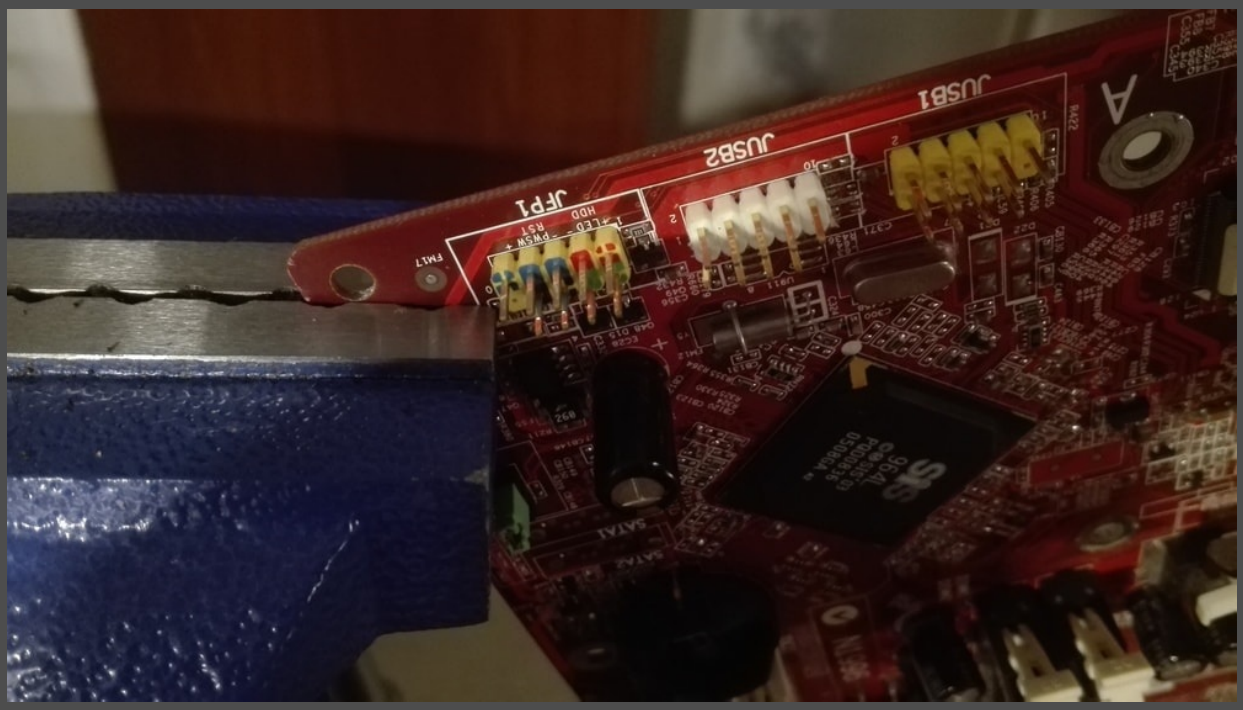
- 16×2 character HD44780 liquid crystal display
- Soldering iron

- Old main board of a desktop computer or pin strip 2,54mm 40 Pin
- Flat nose pliers
- Prototype cable
- Potentiometer (10k)
- Resistor (220 Ohm / red red brown gold)
- Helping Hand
- Breadboard

Preface

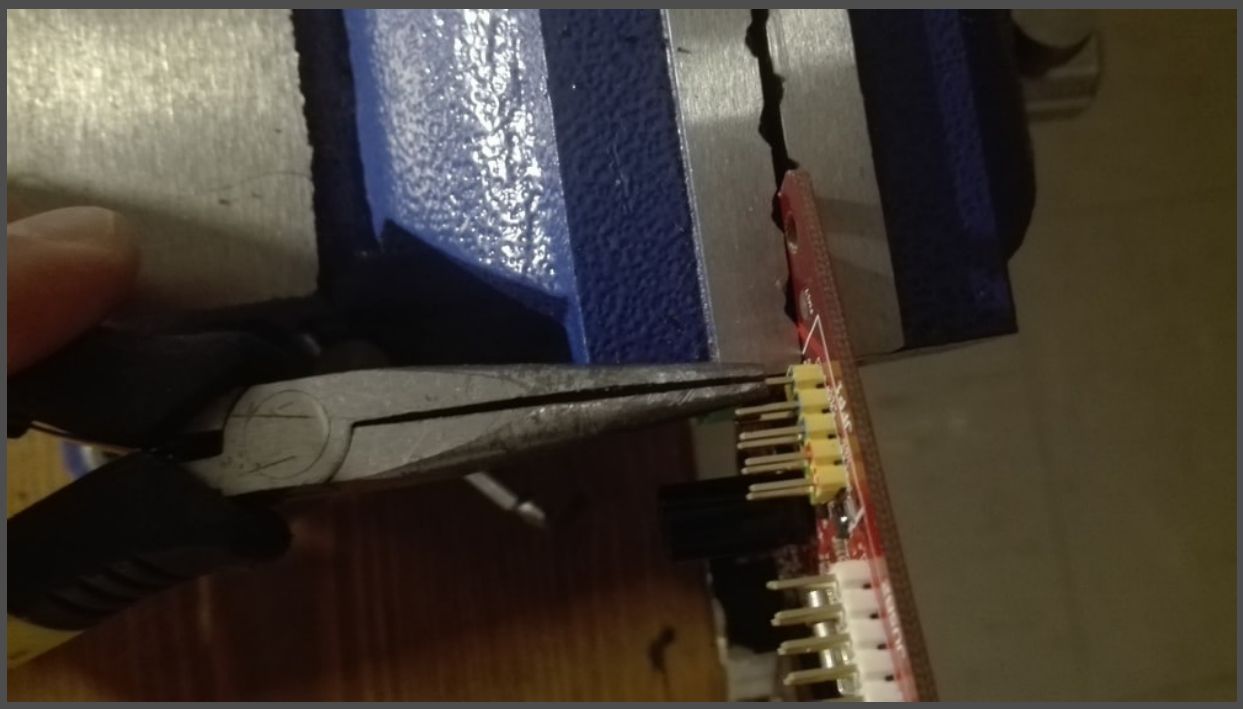


For this documentation I had to order some pinheaders, because I didn't have any more in stock. Unfortunately the delivery from China took me much too long and therefore I looked around for another solution. Desoldering of components is possible, but also quite time-consuming. It also bothers me when too much electricity is consumed.



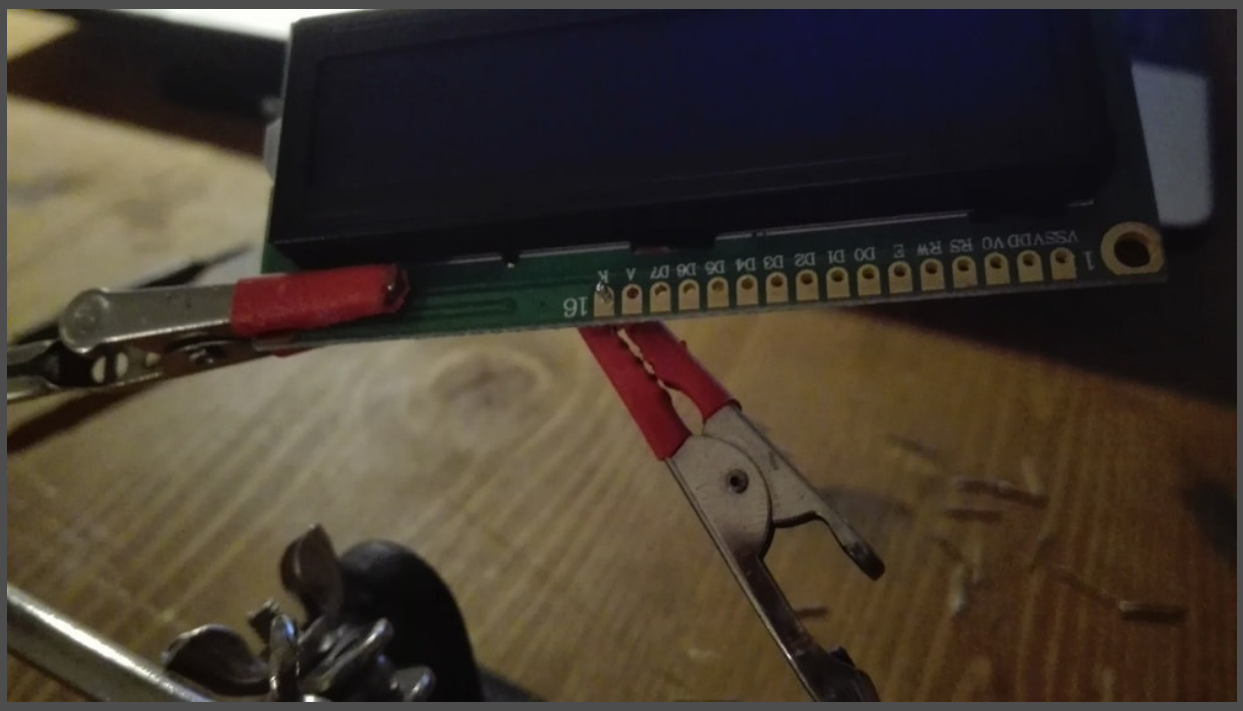
Since the main board is too big for a helping hand, I simply clamped it firmly into the parallel vice. There it cannot slip if we unsolder the tiller with the hot soldering iron

We desolder the tiller by melting the soldering iron on the underside, so that we can pull the tiller or metal pins out of the main board with pliers.



Normally we hold the pliers with one hand and the soldering iron with the other. But since I only have two hands, I still have to take the photo with one. If a pin is bitchy and does not want to be released easily, it sometimes helps to fuse the old spot with new solder and then try to desolder the pin again. In most cases this has helped me quite well.

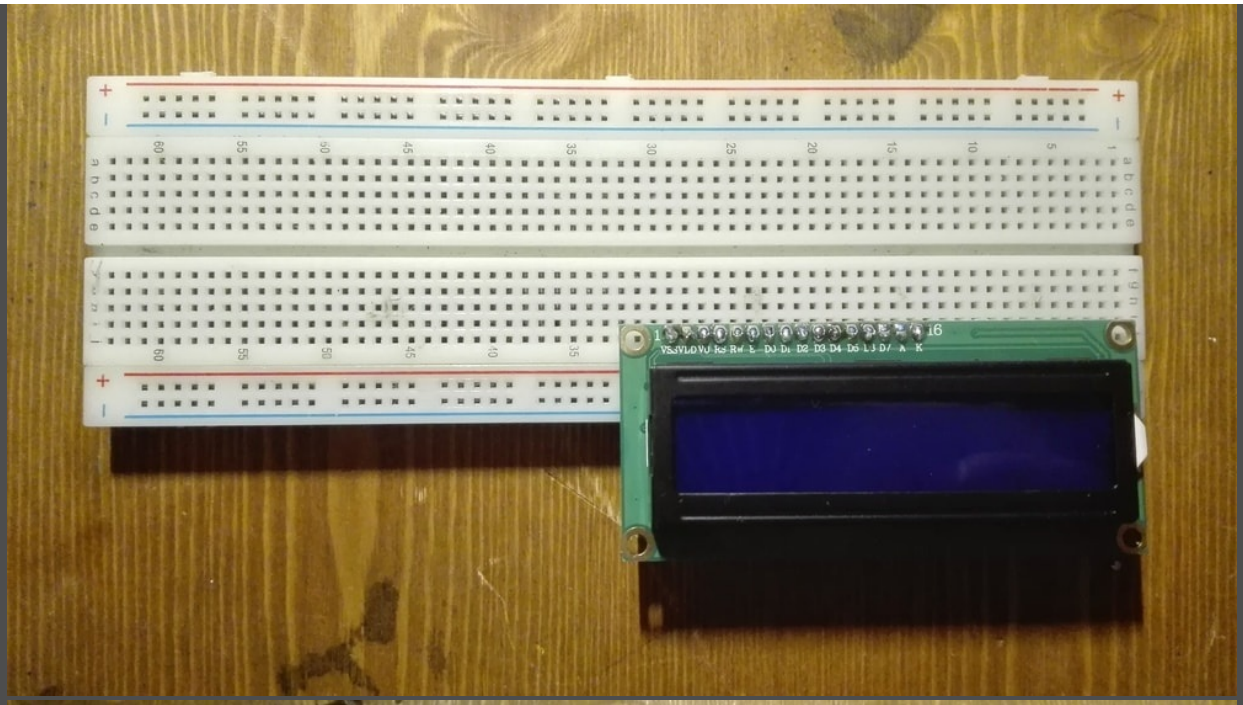
Here we see the desoldered tiller. At the lower ends you can still see the old solder. In the upper right corner we can see a roll of new solder.

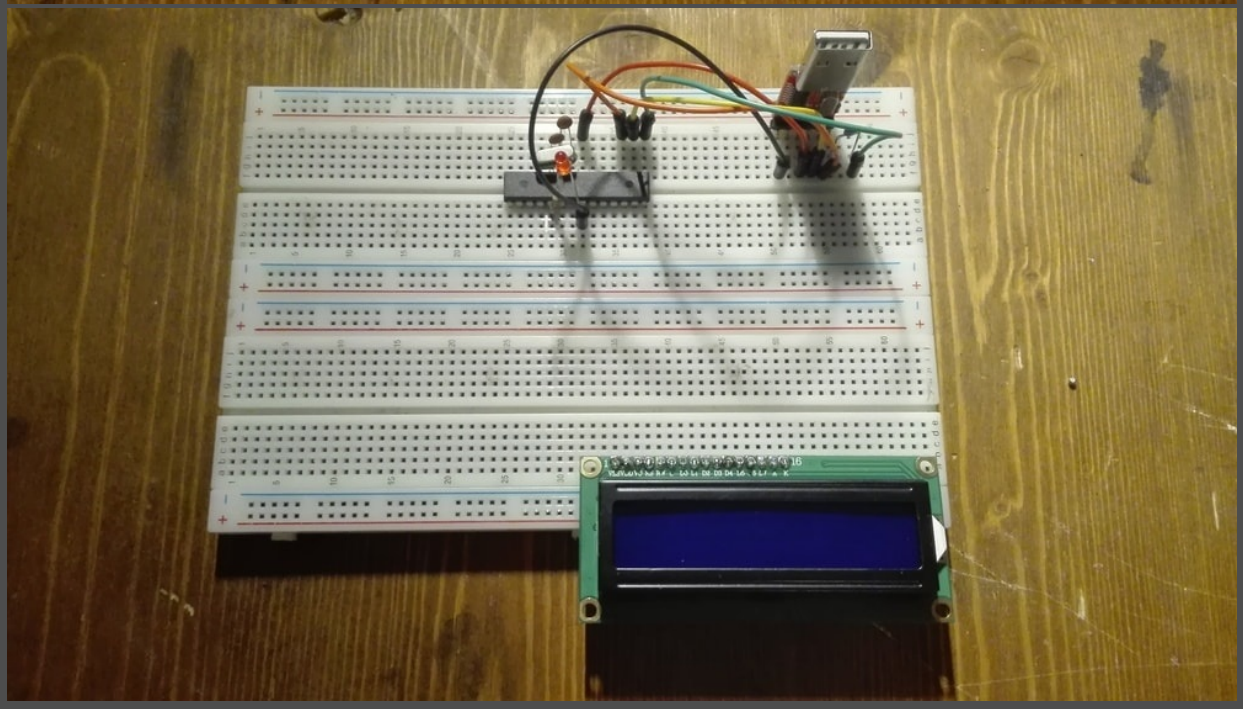


After we have soldered enough tiller from the main board, we clamp the liquid crystal display into our helping hand and solder the first pin. It doesn't matter whether we start left, right or in the middle. Everyone can decide for himself and doesn't influence our working step. The helping hand can turn out to be a very bitchy tool and you need to gain some experience on how best to work with it. By the way, we have to be careful not to break off the resistors on the back of the display.

Before we solder on all the tillers, we test on our pin board whether everything works as we thought it would in theory. Especially when we are soldering something firmly, it is better to have a closer look first and go through the work step in our heads. Soldering is easy, soldering something off is stressful and annoying.

When we're done, our liquid crystal display should look exactly like the picture above. Please pay close attention to the gaps between the solder joints, so that no unwanted connections have formed. In the worst case, they can cause a malfunction and damage your display.

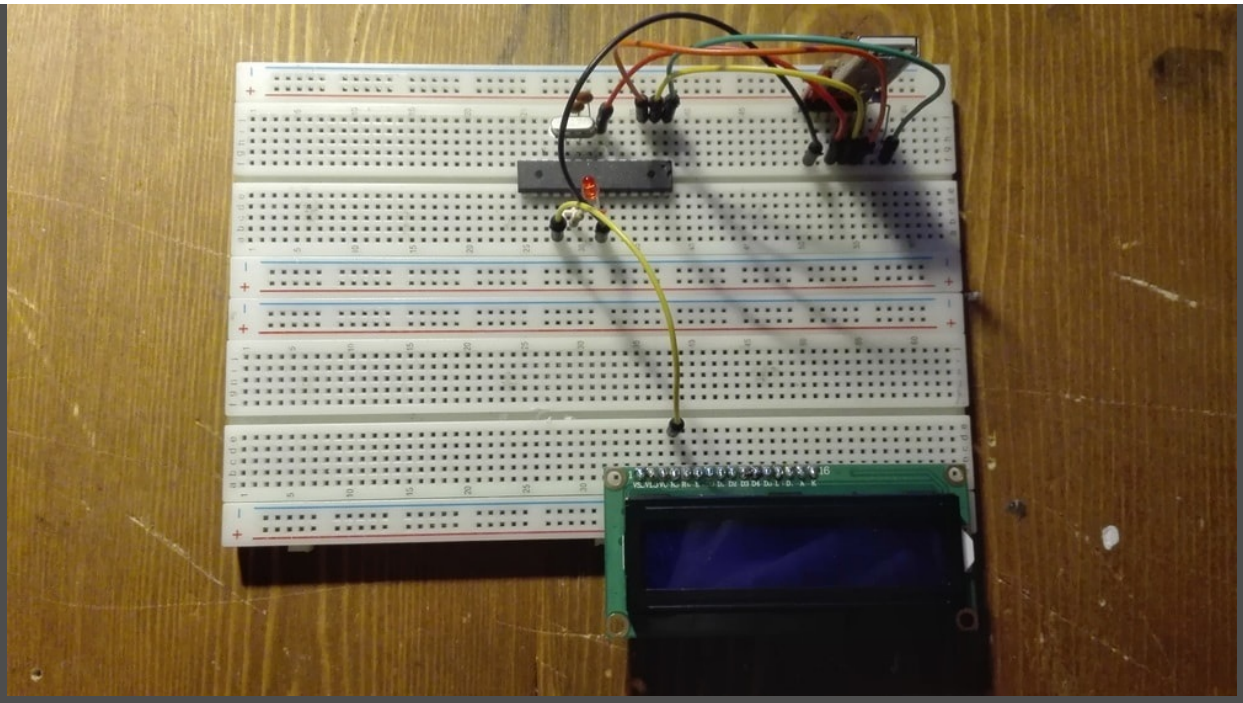




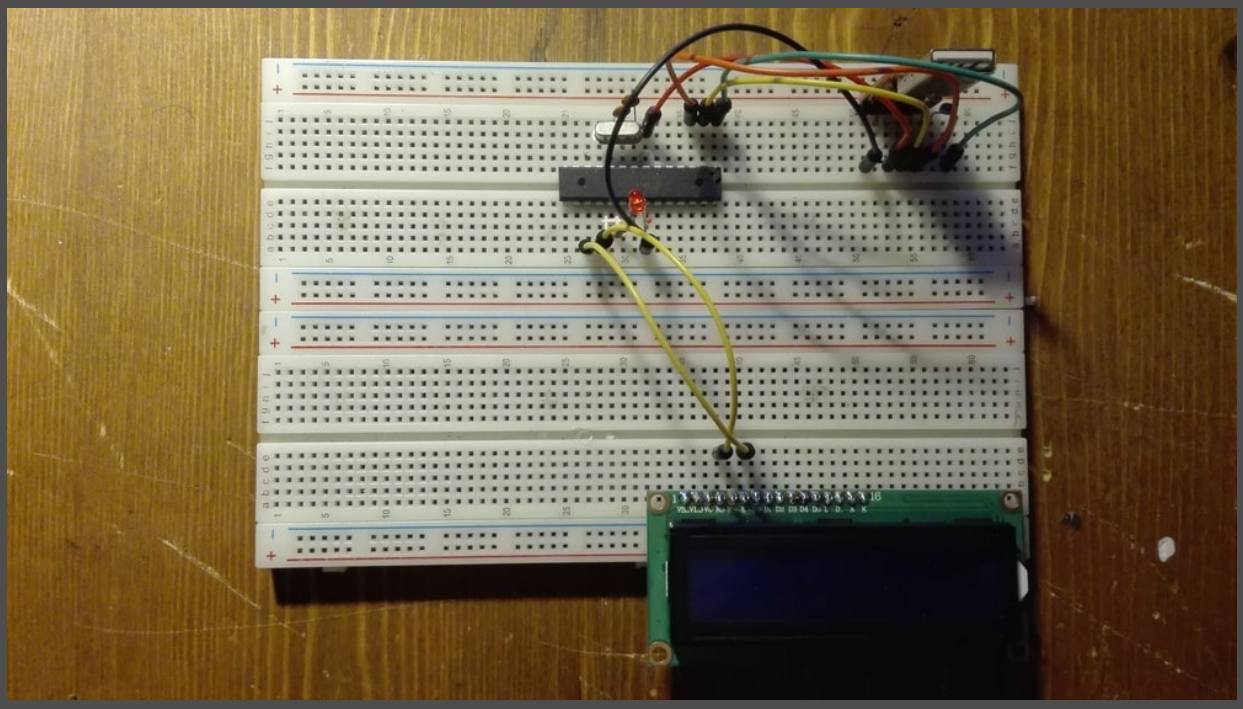
This is our basic structure, with which we can then continue working in peace. Wasn't that difficult so far and there shouldn't have been any relevant problems. At the beginning soldering is always a little difficult and unfortunately you can only learn this through experience, time and constant practice.

Realisation Hardware

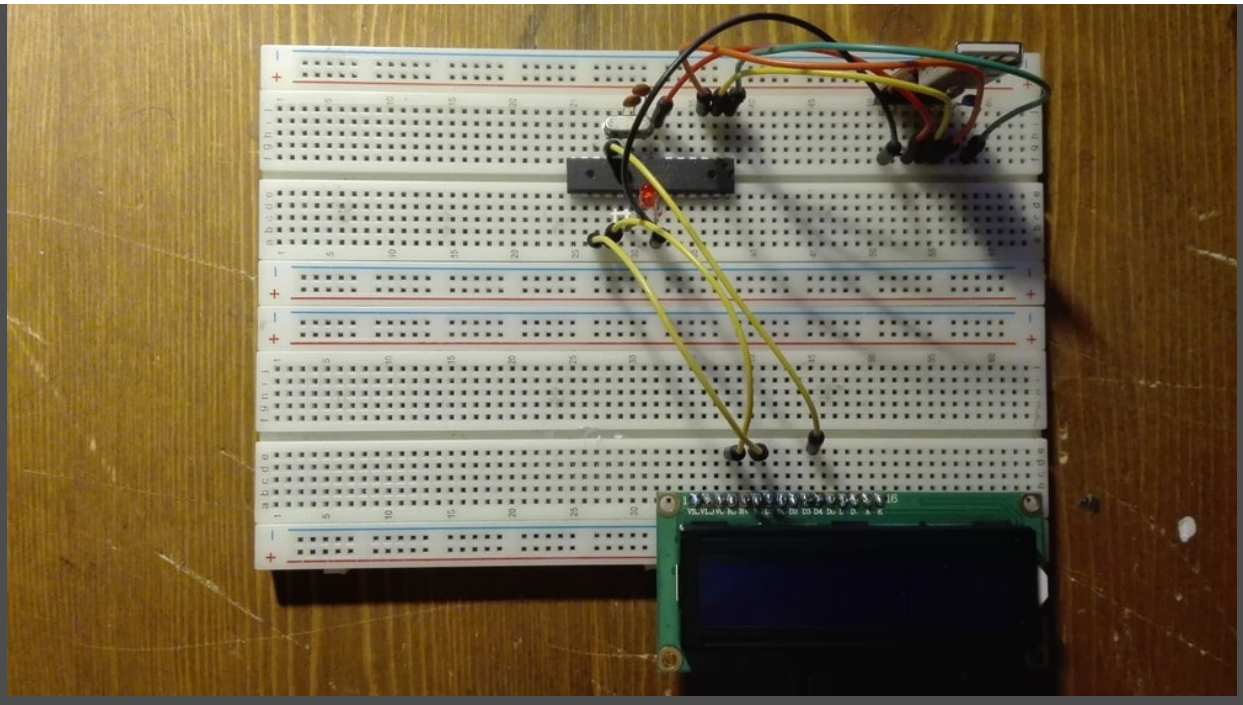
Once again the note: This article sets the cables between the individual tillers differently, since we do not use a purchased Arduino Uno here. I set up and dismantled this circuit a total of three times to test how it all works. This documentation is therefore a special application case. If you are using a purchased Arduino Uno, please follow the tutorial linked above on the Arduino main page.



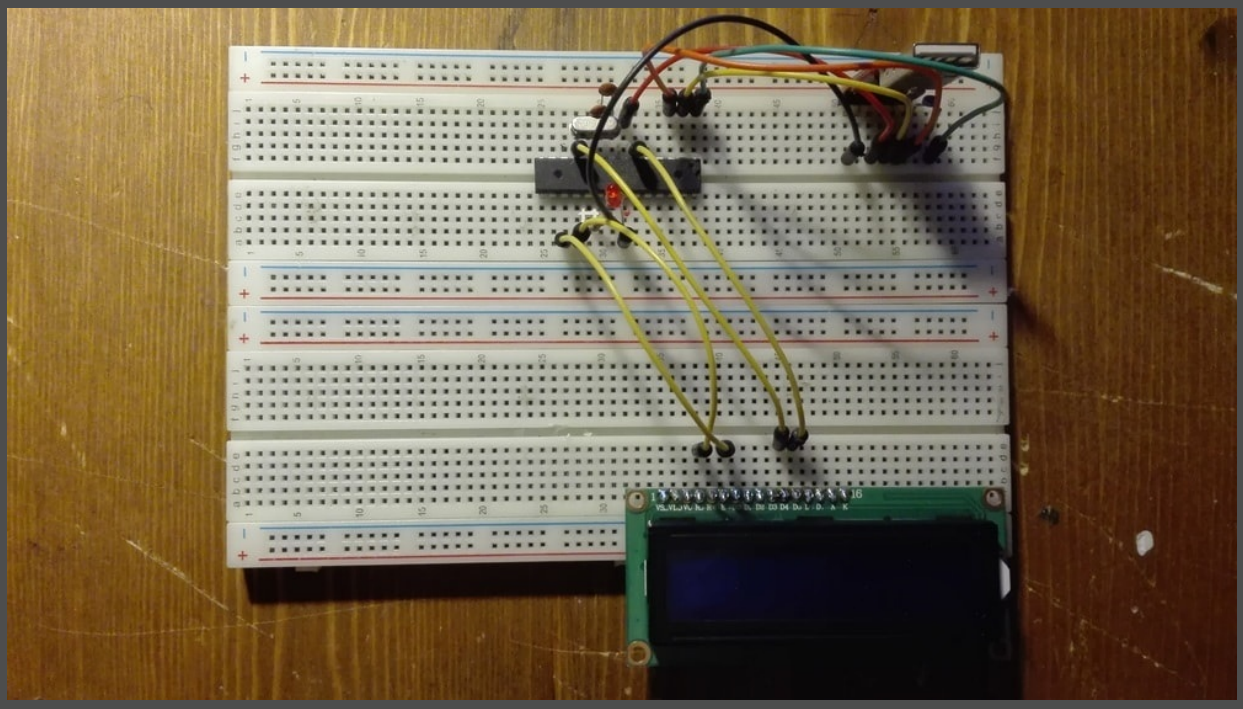
In the first step we connect a yellow cable from `RS` Register Select RS = 0 (instruction register), RS = 1 (data register) to `Pin 18` PB4, (PCINT4/MISO), Digital Pin 12 to our microcontroller. The register selection distinguishes whether instructions or data are sent by us.



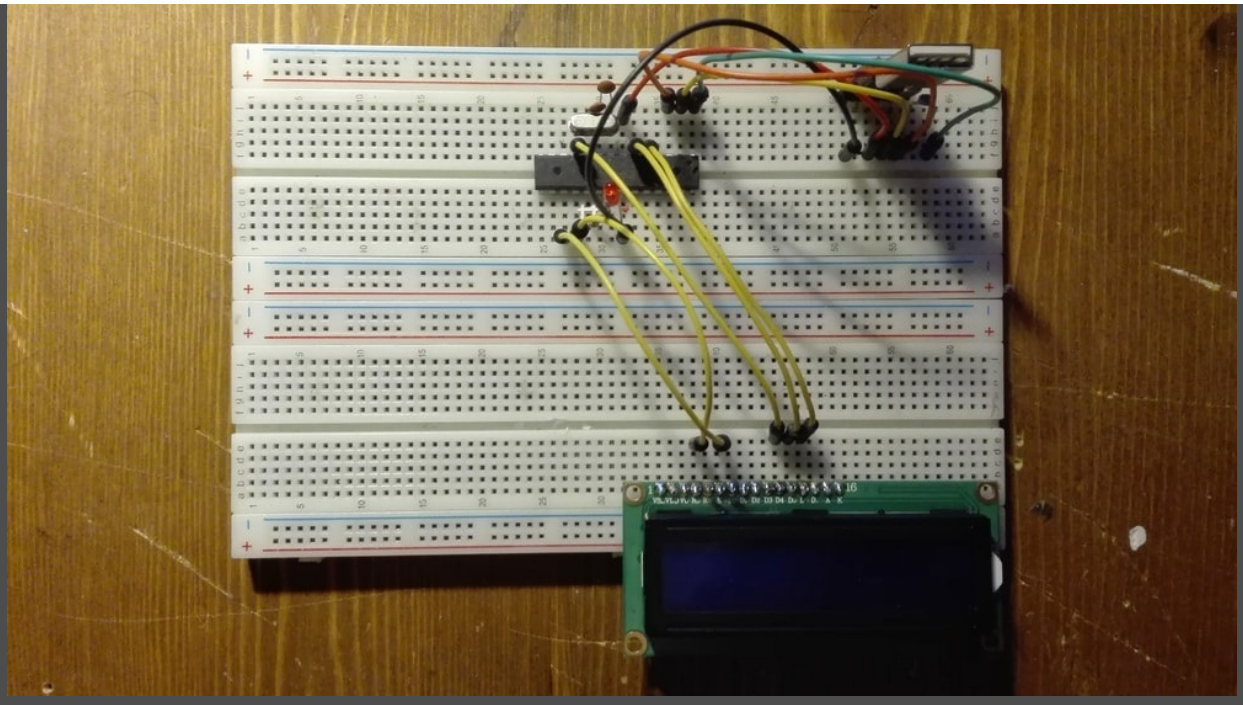
The next cable is connected from `E` (Enable) to `pin 17` PB3, (PCINT3/OC2A/MOSI), Digital Pin 11 (PWM). I wasn't quite sure about the importance of lettering on the liquid crystal display because it may differ on models (which differ in quality). I had read it again in some articles, because it is also sometimes referred to as EN.



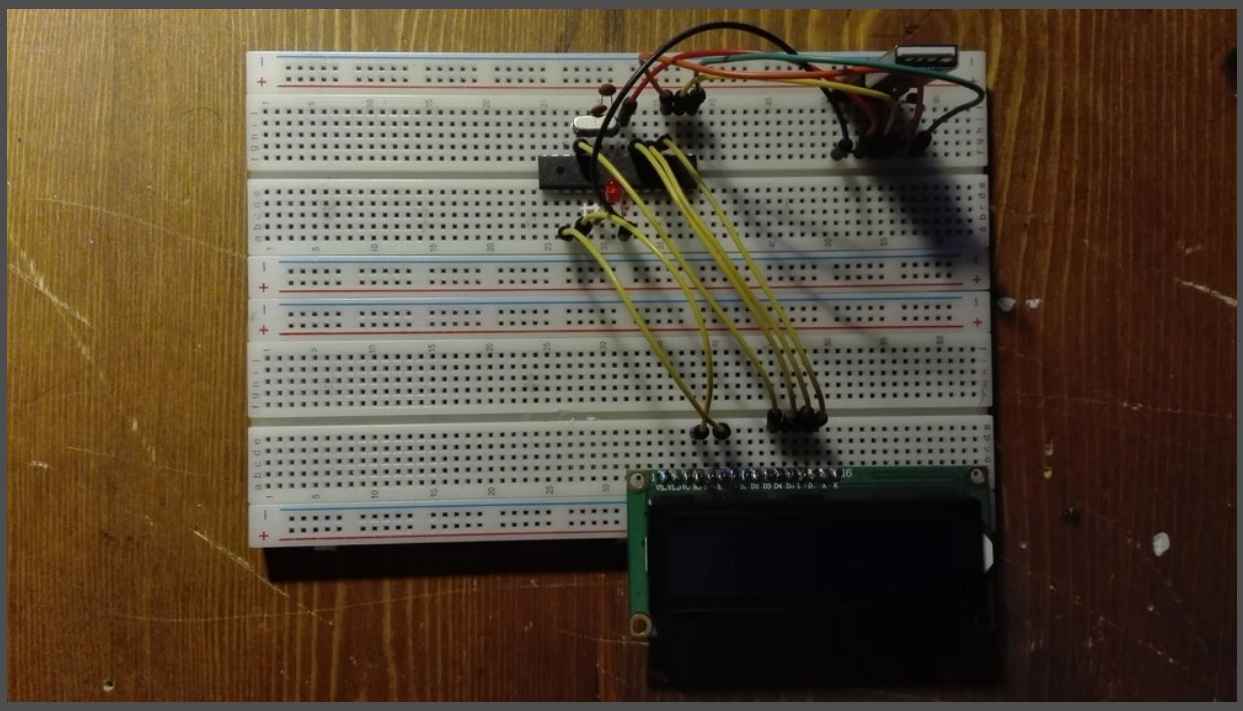
We connect the LCD `D4` data bus to `pin 11` PD5, (PCINT21/OC0B/T1), Digital Pin 5 (PWM). Since we only use a 4-bit instead of an 8-bit data bus, we only have a total of four steps, or cables, that we can use from D4... D7 together.



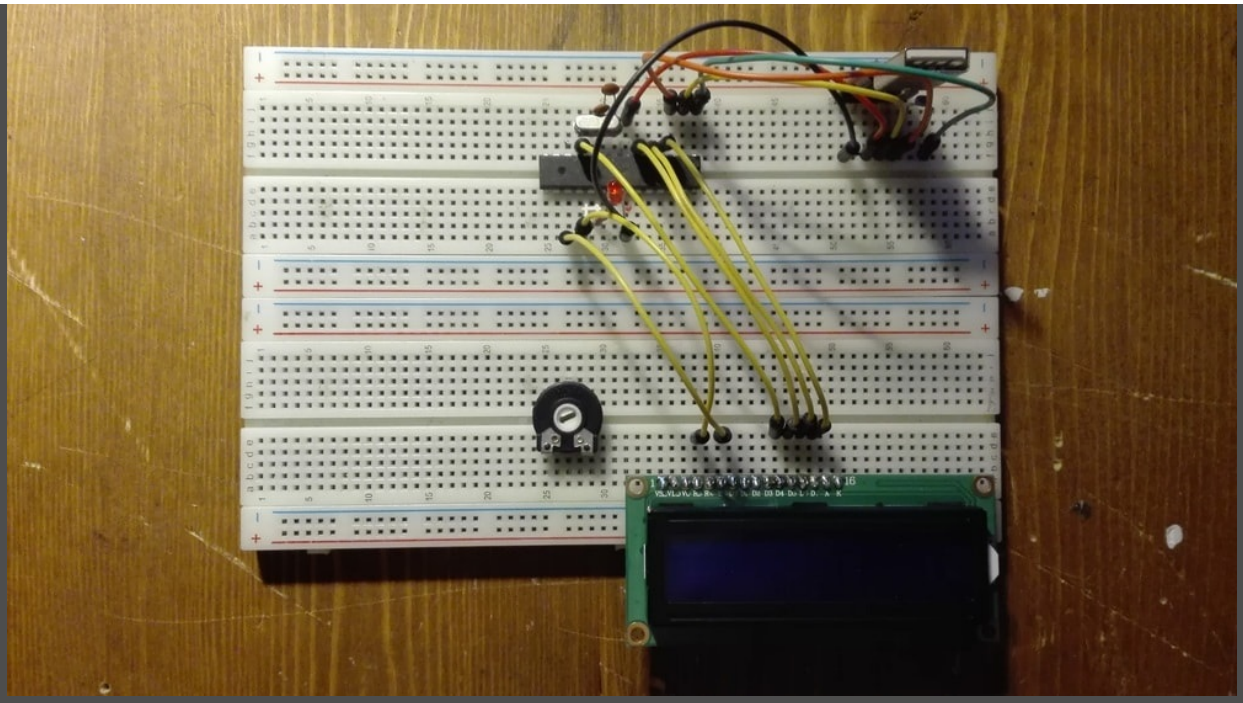
Yellow cable 4 is connected from `D5` to pin `6`



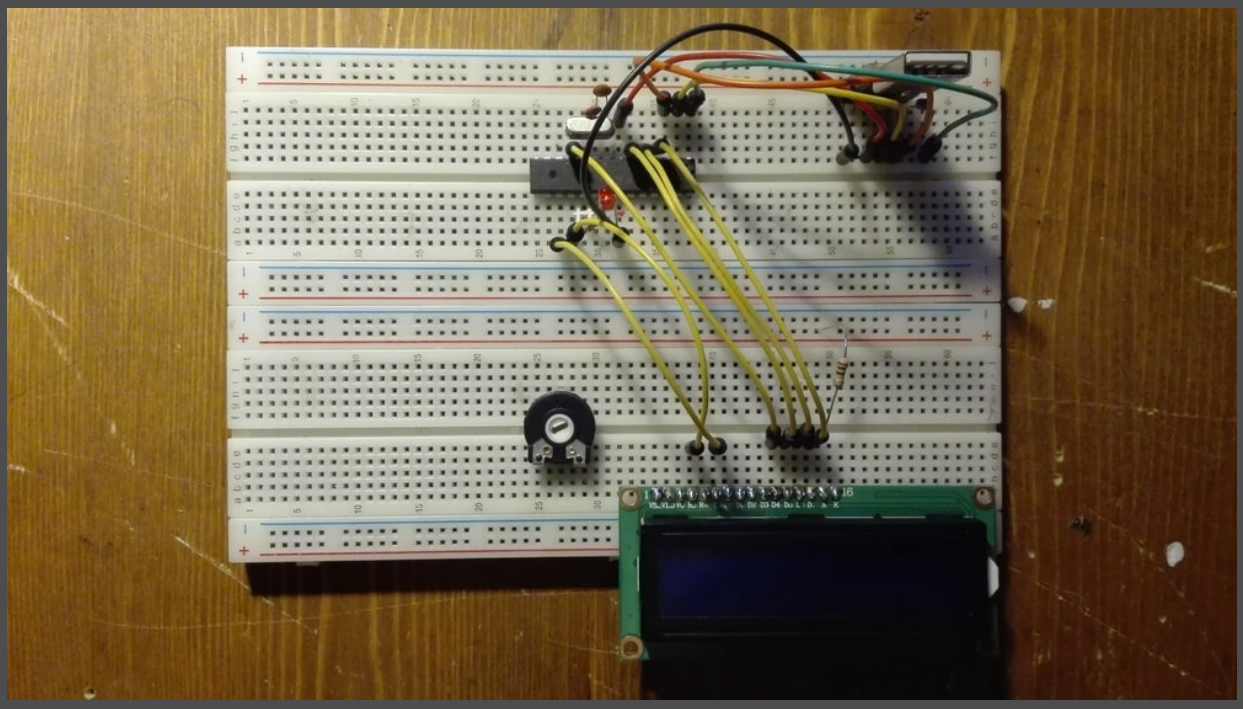
Yellow cable 5 is connected from `D6` to pin `5`



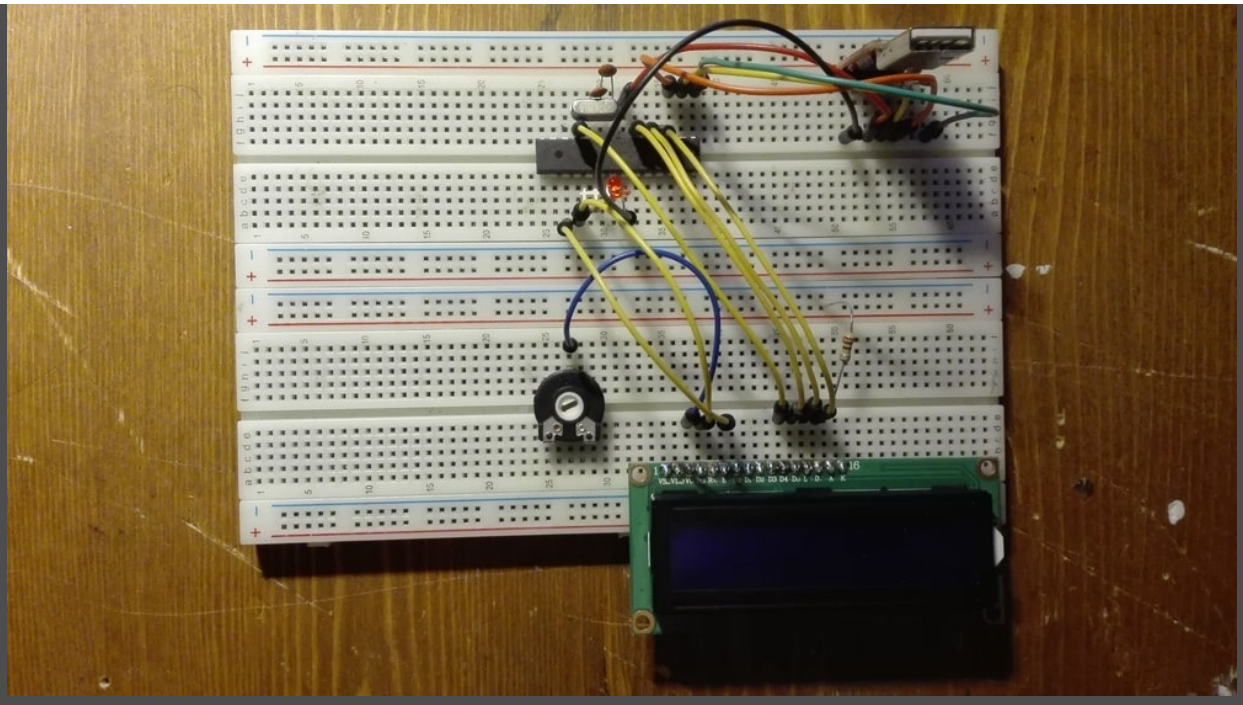
Yellow cable 6 is connected from `D7` to pin `4`



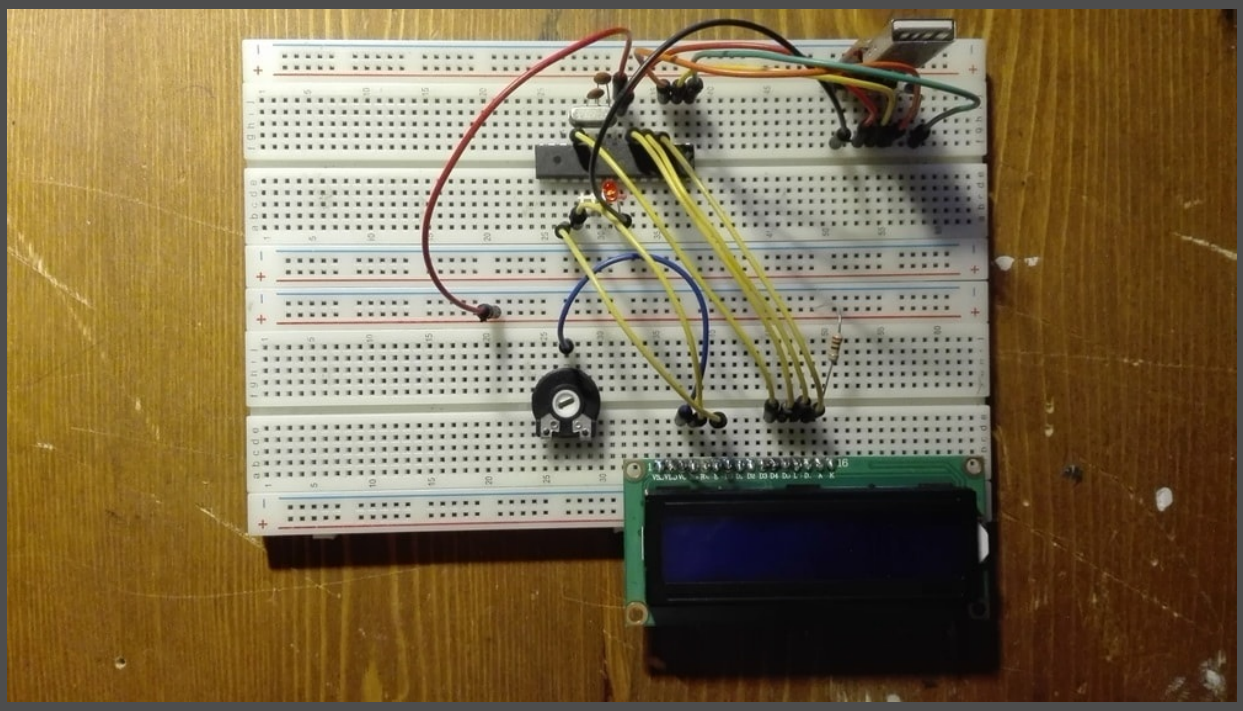
We integrate the potentiometer into our circuit, with which the brightness of the liquid crystal display can be adjusted during the function test. It doesn't matter where exactly the potentiometer is, there should only be enough space around it so we don't build ourselves a rat's nest.



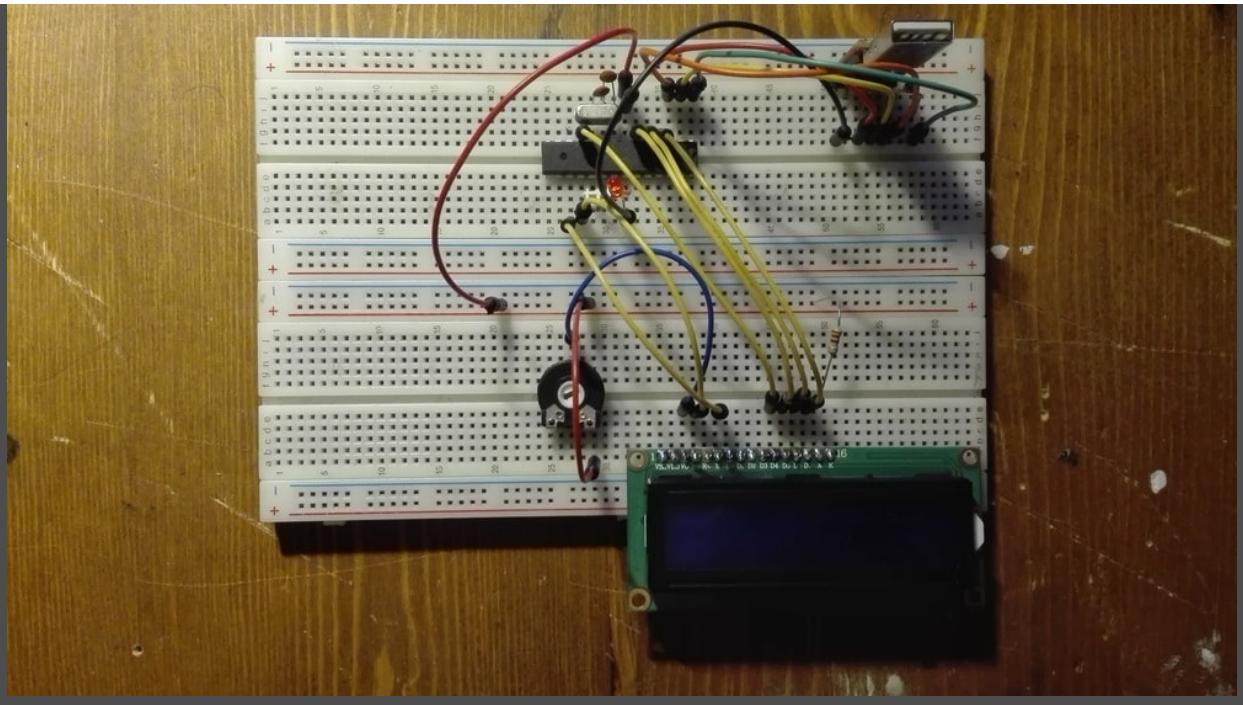
We use the 200kOhm resistor (red, red, brown, gold) in our circuit. This goes from `A` (on the display) to the plus `(+)` line on our pinboard. Plus(+) are the horizontal slots on the pinboard above the red line. Minus(-) is then below the blue line. There are always four horizontal slots on a pin board to help us set up our projects, as we always have to form a closed circuit.



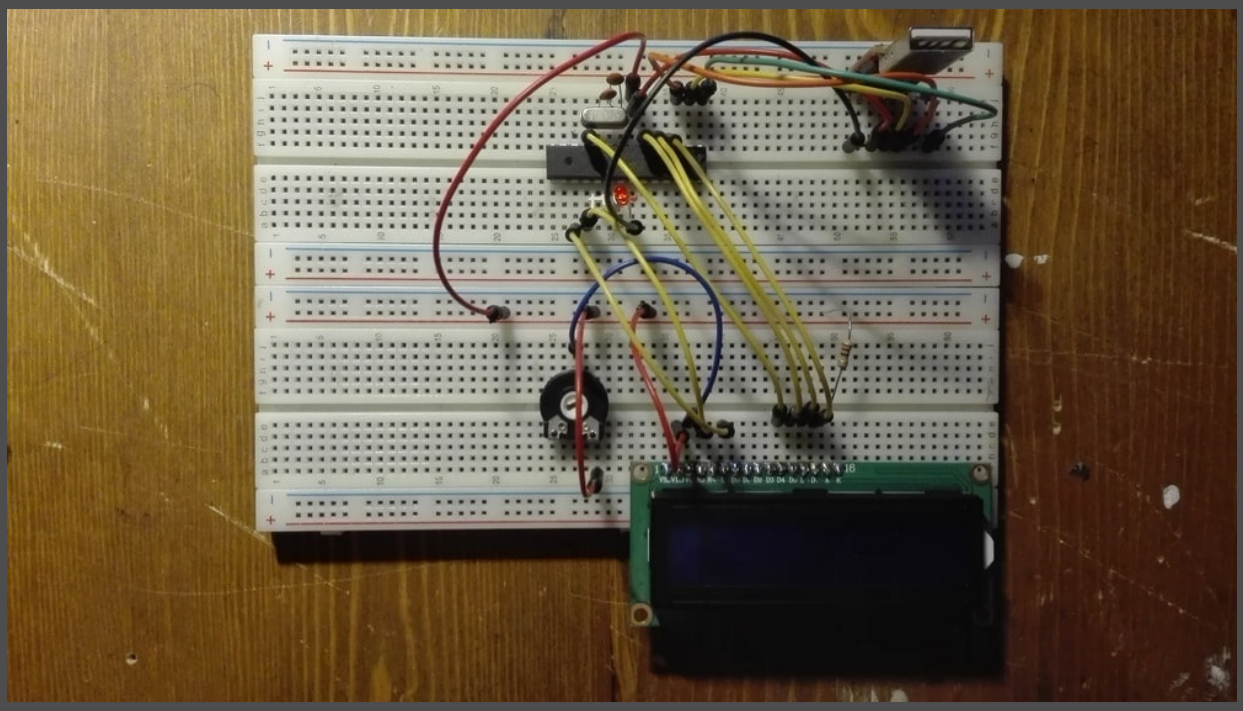
Blue cable from Potientiometer `middle up` to `V0`



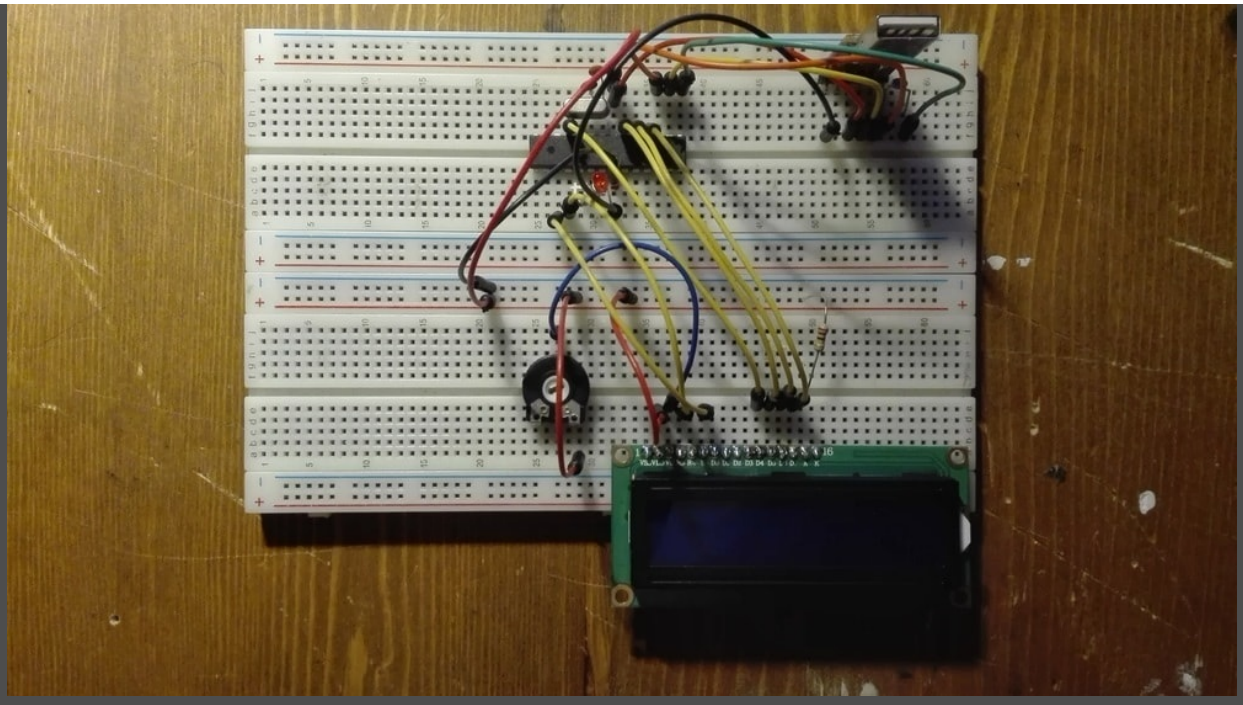
Red cable 1 from `pin 7` to plus `(+)` line on breadboard



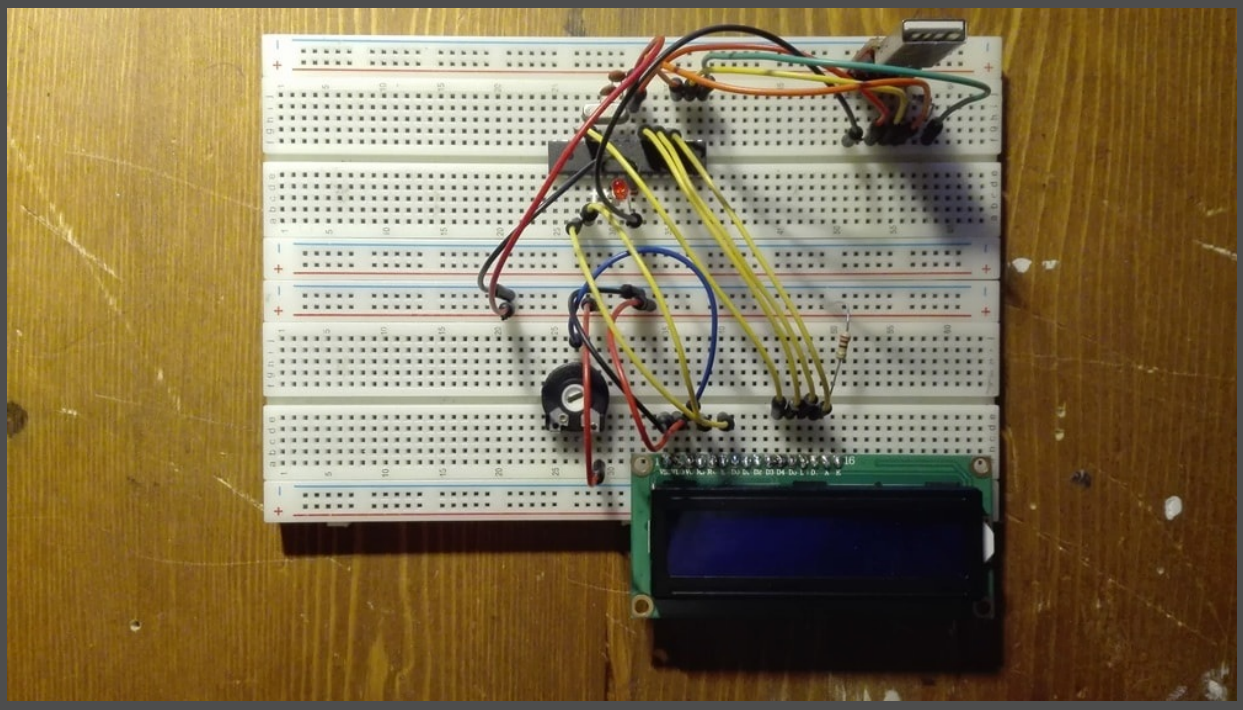
Red cable 2 from plus `(+)` line on pinboard to `potentiometer bottom right`



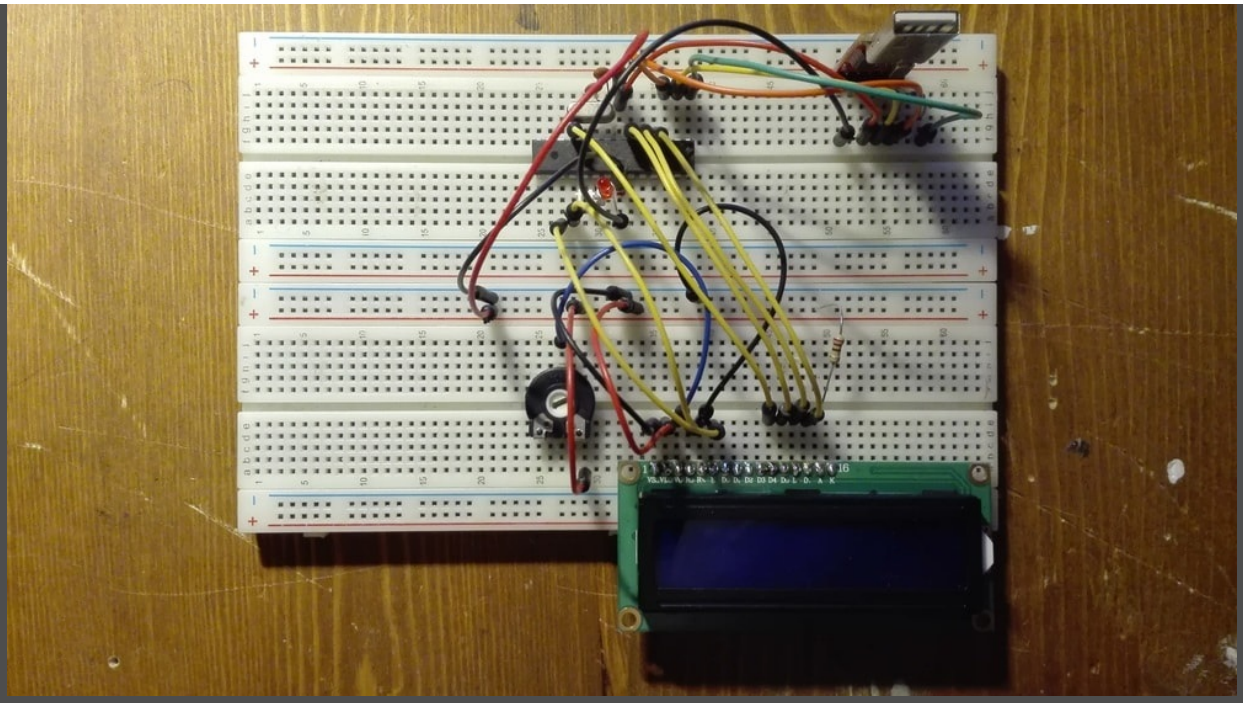
Red cable 3 from plus `(+)` line on pinboard to `VDD`



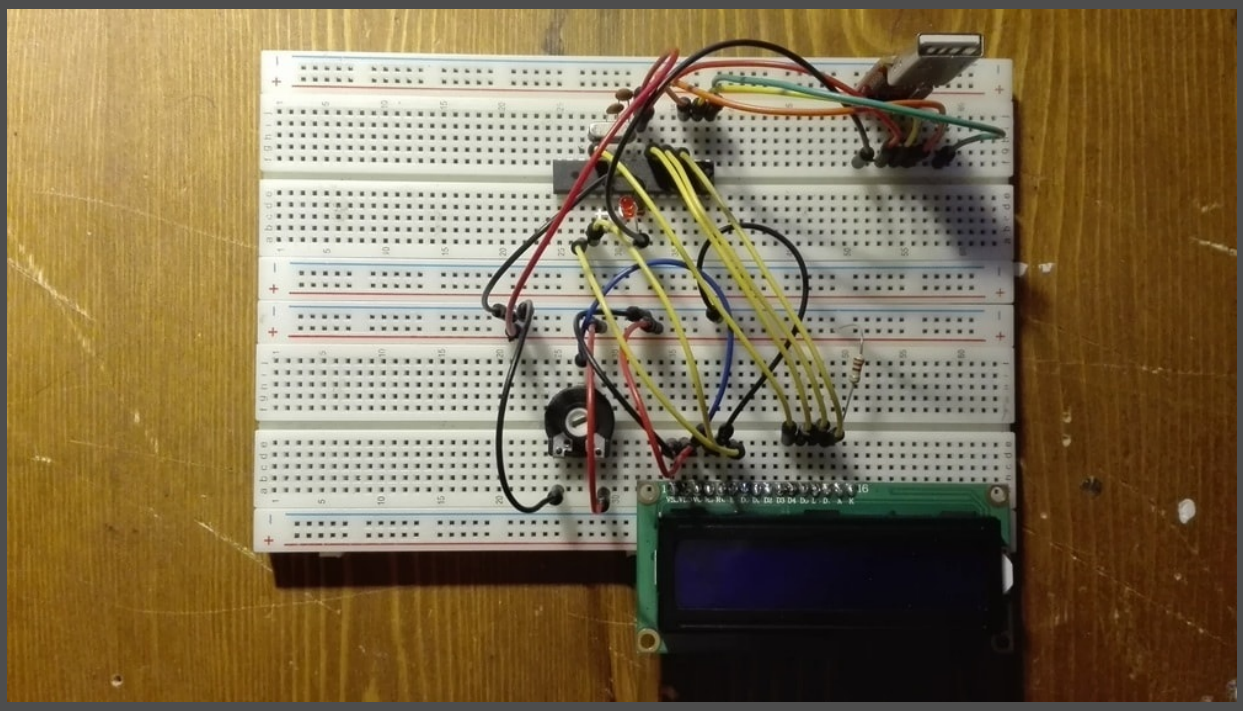
Black cable 1 from `pin 22` to minus `(-)` line. The black and red prototype cables are current-carrying lines with red plus(+) and black minus(-) defined. You should keep this in mind, because if we cut out a USB cable, we can see the two colors mentioned above between the data cables (white, green). Believe me. There are reasons why I rebuilt and dismantled the circuit three times.



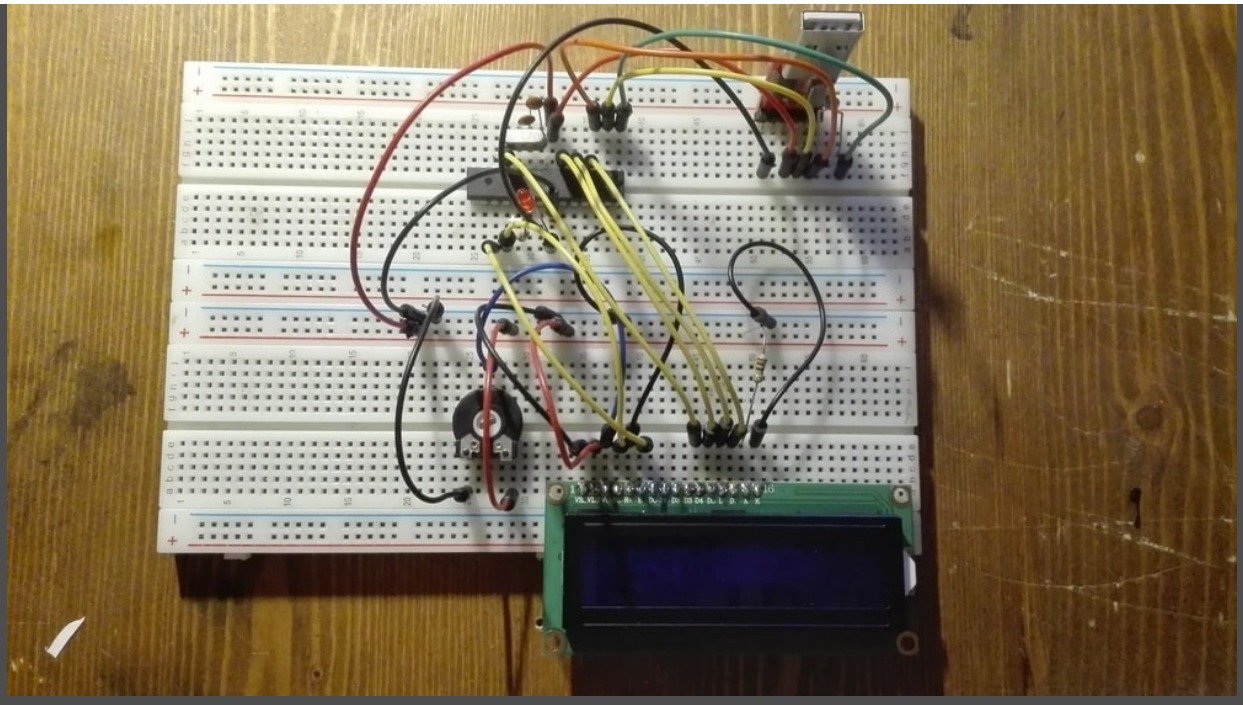
Black cable 2 from Minus `(-)` line on breadboard to `VSS`



Black cable 3 from Minus `(-)` line on pinboard to `RW`



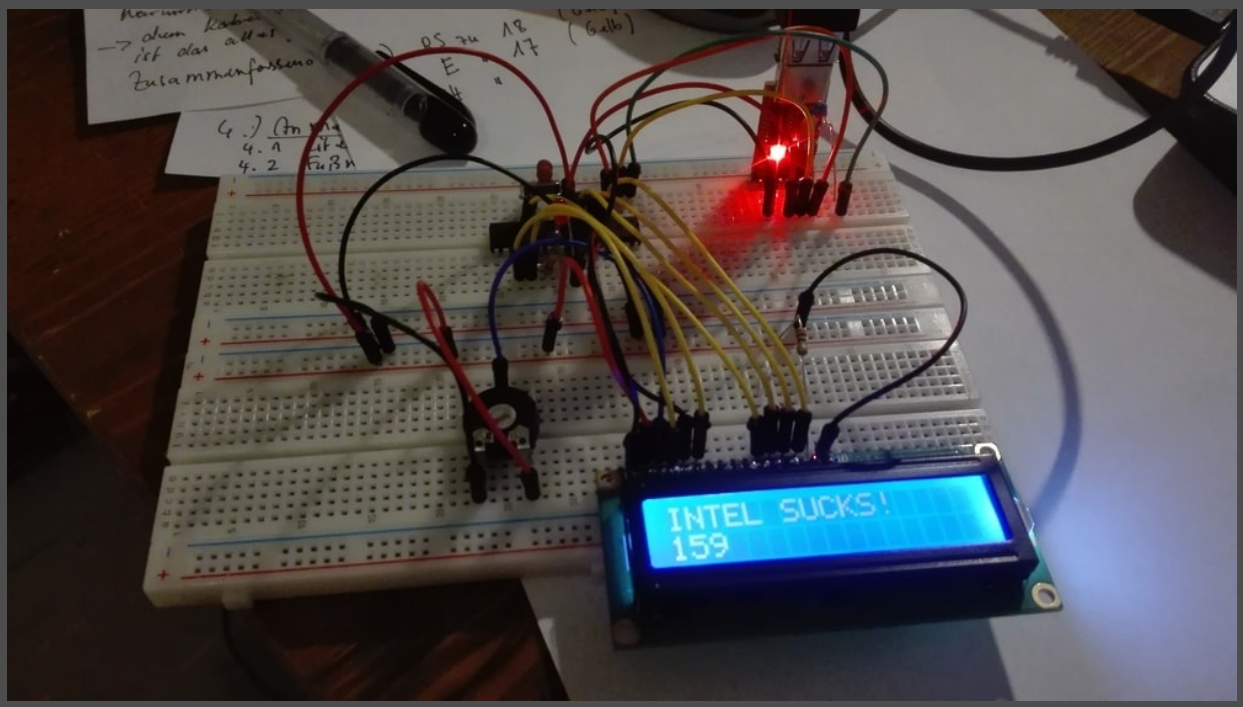
Black cable 4 from minus `(-)` line on breadboard to `potentiometer bottom left`



Black cable 5 from minus `(-)` line on breadboard to `K`. Thus our circuit is finished from the hardware side. And here's another piece of advice. Work as cleanly as possible, as this saves you time in troubleshooting.

Realisation Software

As already explained in the keyboard controller documentation, load the source code LCD.txt into our Arduino IDE and transfer it to our microcontroller. After we have transferred our software to the microcontroller, our LED flashes several times and the text entered by us is displayed. Besides, our circuit counts up from zero as long as it is switched on.



Some potentiometers do not have a small knob, in this case you take a slotted screwdriver as you can see at the top of the picture and insert it carefully and can adjust the potin. A toothpick is incredibly practical for rough hands and helps to count the individual slots on the pinboard. It can also be inserted somewhere as a memory aid and it is very difficult to trigger a short circuit (because it is made of dry wood).

Troubleshooting

If you only see black boxes on the liquid crystal display, you have placed one or more cables incorrectly. So you have to check your whole setup again and see exactly where the error is. This only happens if you have worked uncleanly. In this documentation we assume that we have not changed the source code and that the display has been bought and tested again.

Conclusion

Once we have connected our liquid crystal display to our project, we need to think about how we can use it sensibly. I imagine that this could be used as a source of error output. Although there are also some projects on the net that also use the display as a screen for very simple computers. I don't know if I really want to. I still have some thoughts and ideas that I want to try out.